

C# 프로그래밍

- 1주차 (1강)

강의 소개

- 교과 목표
- 교재
- 강의 목차

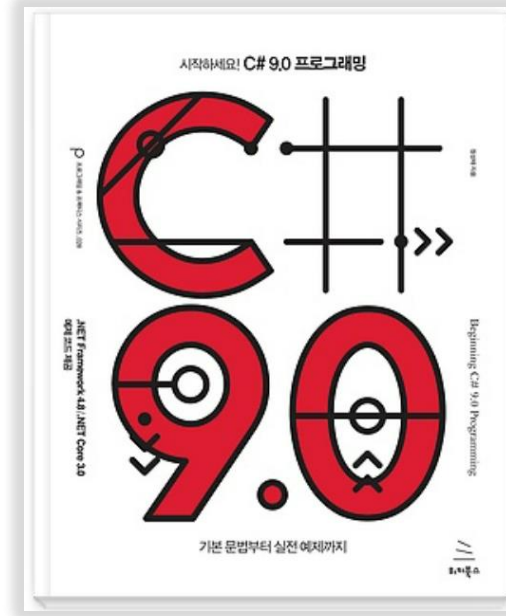
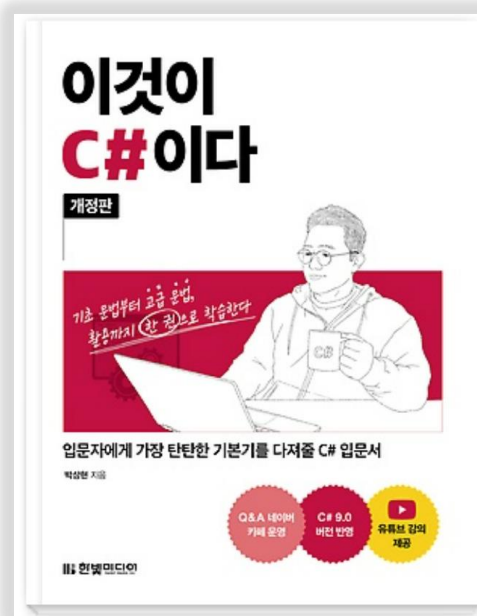
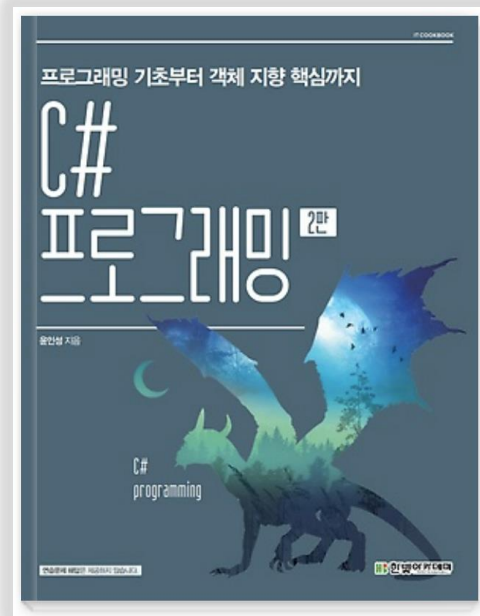
교과 목표

- C# 프로그래밍 언어의 기본 및 고급 문법 학습
- 다양한 예제코드와 함께 각 문법의 활용법 습득
- 객체지향 프로그래밍의 충분한 이해
- .NET 클래스 라이브러리의 활용
- C# 언어를 이용한 프로그램 구현 실습



주 교재

- C# 프로그래밍 (개정판 2021), 윤인성 지음, 한빛아카데미
- 이것이 C#이다 (개정판 2021), 박상현 지음, 한빛미디어
- 시작하세요! C# 9.0 프로그래밍, 정성태 지음, 위키북스



강의 목차 (1)

주별 (Week)	내용	수업형태
1	1강: C# 언어와 .NET 플랫폼 소개 2강: C# 기본 문법 - 데이터 타입, 변수, 문자열 (1)	이론: 온라인 동영상
2	1강: C# 기본 문법 - 데이터 타입, 변수, 문자열 (2) 2강: C# 기본 문법 - 연산자	이론: 온라인 동영상
3	코드 흐름 제어 - 조건문, 반복문, 점프문	이론: 온라인 동영상 실습: 웹엑스
4	클래스 기본 - 클래스의 개요, 사용, 생성, 추상화	이론: 온라인 동영상
5	메서드 - 기본 형태, 클래스 메서드, 오버로딩	이론: 온라인 동영상
6	메서드 - 접근 제한자, 생성자/소멸자, 속성, 값/참조 복사	이론: 온라인 동영상
7	상속과 다형성 - 소개, is 키워드, 새도잉, 하이딩, 오버라이딩	이론: 온라인 동영상 실습: 웹엑스
8	중간고사	



강의 목차 (2)

주별 (Week)	내용	수업형태
9	인터페이스 – 소개, 생성, 멤버, 다중 상속	이론: 온라인 동영상
10	클래스 심화 – 배열, 인덱서	이론: 온라인 동영상
11	클래스 심화 – 일반화 프로그래밍 (제너릭)	이론: 온라인 동영상
12	윈도폼 기본 익히기 – 메서드, 상속, 다형성	이론: 온라인 동영상 실습: 웹엑스
13	예외 처리 개념 및 구분	이론: 온라인 동영상
14	델리게이터와 람다	이론: 온라인 동영상
15	응용예제 연습 및 윈도폼 활용	이론: 온라인 동영상
16	기말고사	



C# 프로그래밍 시작하기

- C# 과 .NET 플랫폼
- 개발 도구 설치
- 예제 코드

C# 프로그래밍 시작하기

- C# 과 .NET 플랫폼

- 개발 도구 설치

- 예제 코드

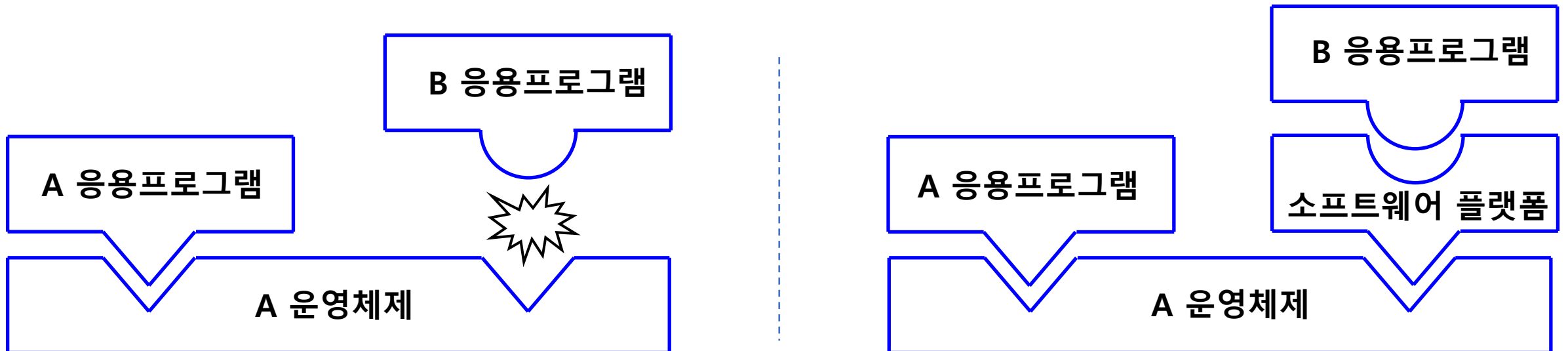
C# 프로그래밍 언어

- Microsoft 사의 앤더스 헤일스버그 (Anders Hejlsberg) 개발
- .NET Framework에 최적화
- 객체지향 (Objected-Oriented) 언어
- 다양한 플랫폼에서 동작
 - ✓ 윈도우, 맥, 리눅스, 안드로이드, 아이폰 등
- 프로그래머에게 높은 수준의 생산성 제공



.NET 플랫폼

- 플랫폼 (Platform) 이란?
 - ✓ 응용프로그램을 실행하는 데 사용되는 하드웨어와 소프트웨어 집합
 - ✓ 예: 윈도, 맥, 안드로이드, 아이폰(iOS) 등등
- 소프트웨어 플랫폼
 - ✓ 운영체제 차이에 따른 응용프로그램 실행환경 문제 해결



CLR (Common Language Runtime)

- .NET 라이브러리와 함께 운영체제 위에 설치
- 적시 (Just In Time) 컴파일
 - ✓ 소스코드 -> 중간언어 (IL: Intermediate Language) 코드 -> 네이티브 코드
 - ✓ C# 외에 다양한 프로그래밍 언어 지원
 - ✓ 장점: 플랫폼에 최적화된 코드 생성
 - ✓ 단점: 실행 시 컴파일 부담

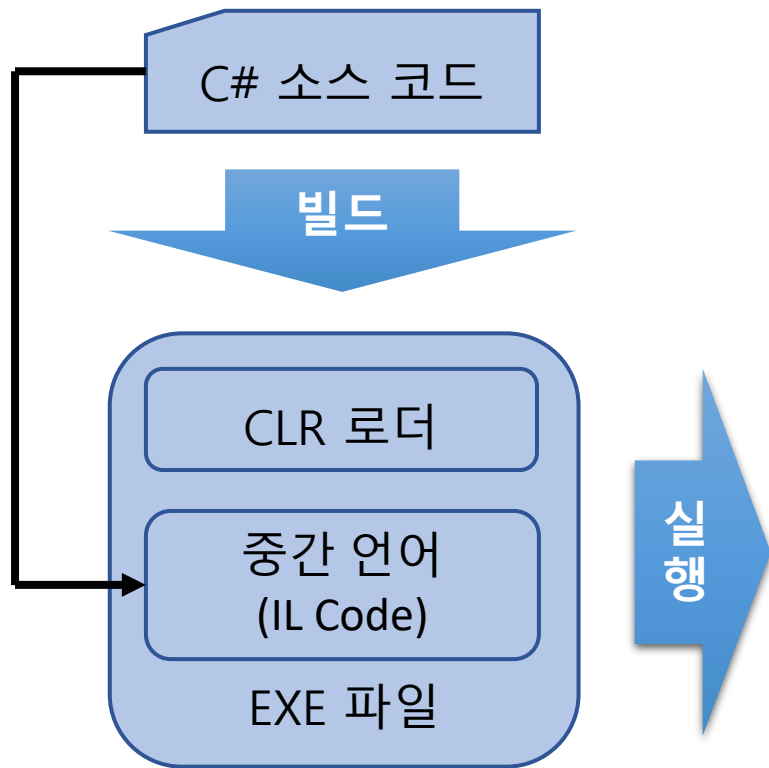
.NET Framework

C# 응용프로그램	VB.NET 응용프로그램	C++ 응용프로그램	등등
.NET 라이브러리 Common Language Runtime (.NET 플랫폼)			
운영체제 (윈도, 리눅스)			
하드웨어 (x86/x64, ARM, ...)			



IL 코드 실행하는 CLR

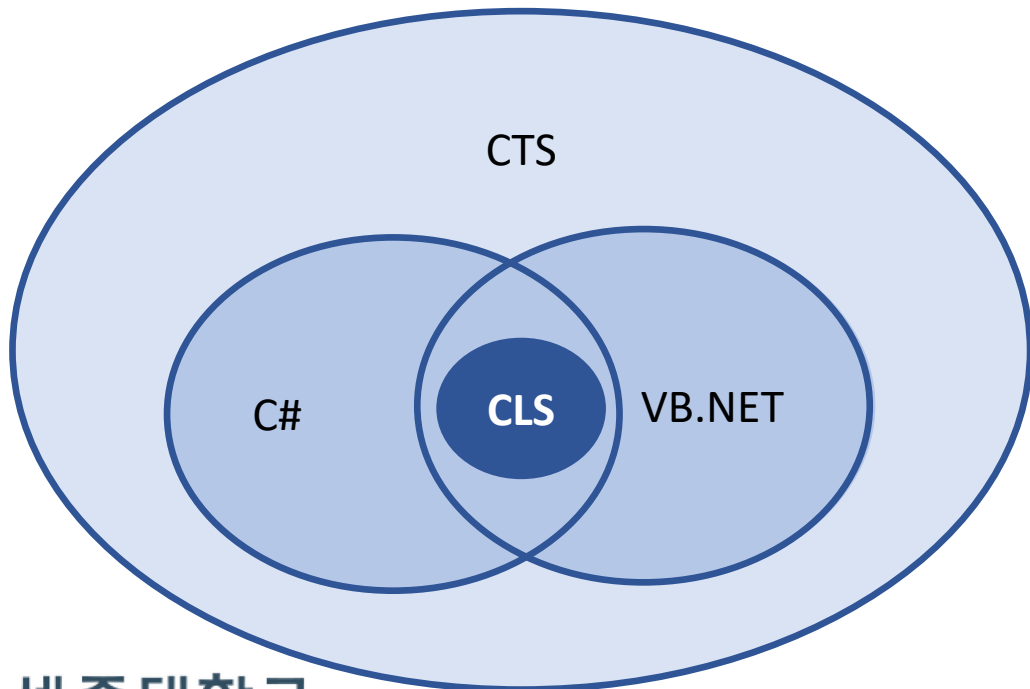
- C# 컴파일러는 소스코드에 대한 EXE 파일 생성



- 생성된 EXE 파일
 - ✓ CLR을 메모리에 불러오는 코드 포함
 - ✓ 중간언어(IL)로 변환된 소스코드 포함
- 프로세스 (EXE 파일) 실행 시
 - ✓ CLR을 메모리에 적재
 - ✓ CLR이 중간언어로 된 .NET 코드를 실행

.NET 호환 언어

- 중간언어(IL)로 변환되는 언어
 - ✓ 예 : C#, Visual Basic .NET, F#, C++ 등등
- 호환 언어는 중간언어로 변환되어 상호 호출이 가능
 - ✓ 메소드, 클래스 등을 공유 가능



- 공용 타입 시스템 (CTS: Common Type System)
 - ✓ .NET 호환 언어는 CTS 규약을 만족하는 한도 내에 구현 가능
 - ✓ .NET 호환 언어가 CTS에서 정의된 모든 규격을 구현할 필요는 없음
- 공용 언어 사양 (CLS: Common Language Specification)
 - ✓ .NET 호환 언어가 지켜야 할 최소한의 언어 사양



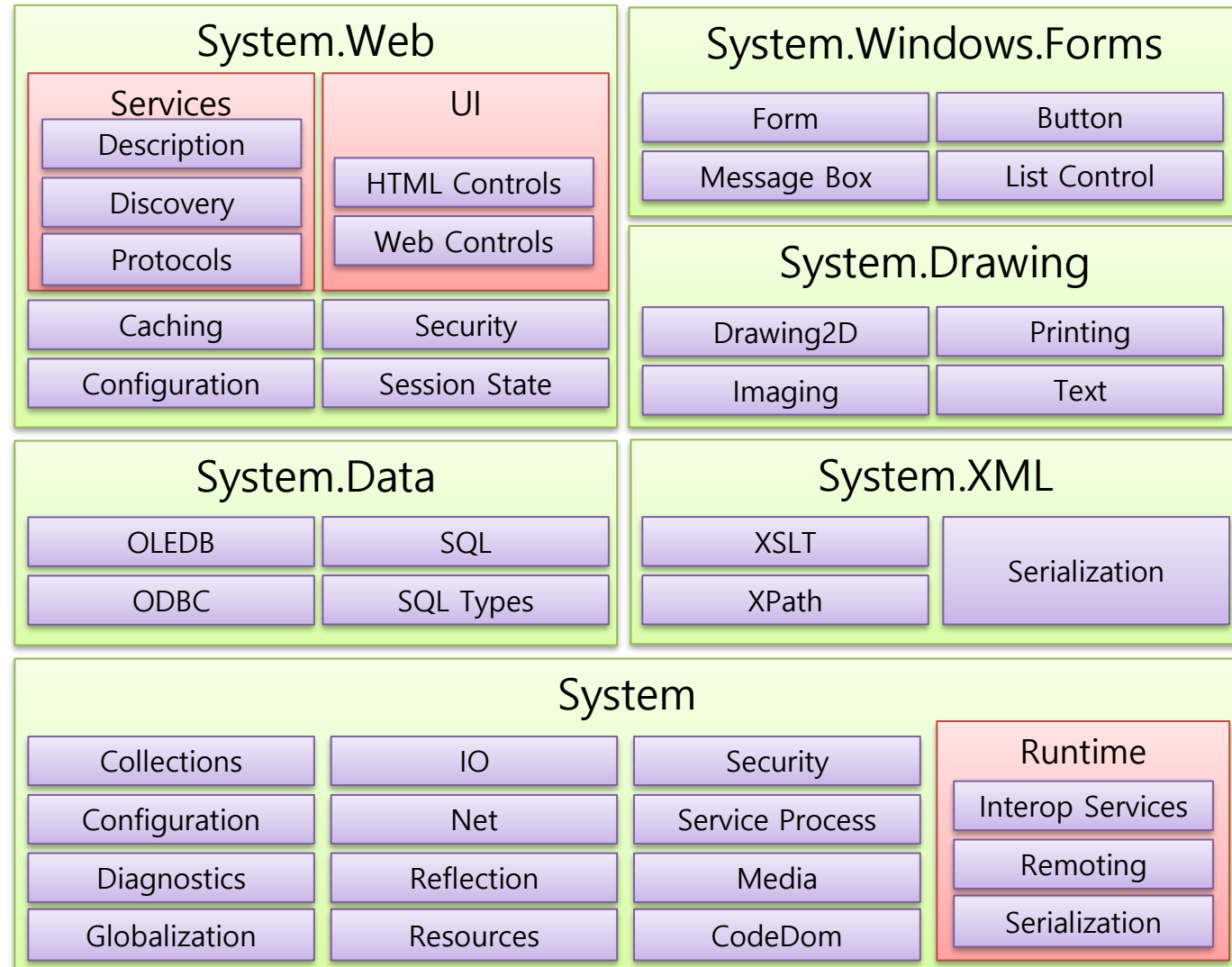
.NET 라이브러리 (1)

- 라이브러리(Library) 이란?
 - ✓ 응용프로그램을 만들 때 사용하는 미리 정의한 클래스 또는 서브루틴의 집합
 - ✓ 예: 화면에 글자 출력을 위한 메서드
 - `print ("Hello World")`
 - `printf ("Hello World")`
 - `Console.WriteLine ("Hello World")`
 - `System.out.println ("Hello World")`
 - `puts ("Hello World")`
- .NET 클래스 라이브러리
 - ✓ 응용프로그램 개발을 위해 정의된 클래스, 네임스페이스, 인터페이스 집합



.NET 라이브러리 (2)

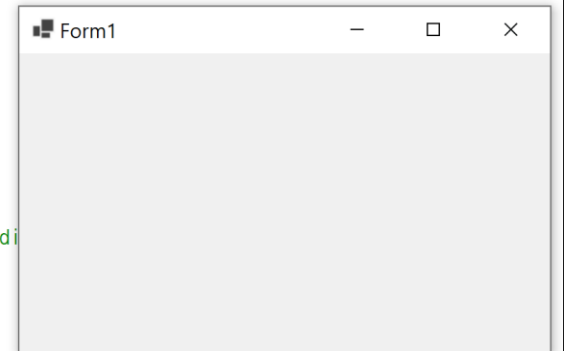
- 네임스페이스 (Namespace)
 - ✓ 용도별/분야별 비슷한 라이브러리 코드를 묶음
- 예: System.IO 네임스페이스
 - ✓ 파일 입출력을 다루는 각종 클래스, 구조체 등의 모음



.NET Framework의 프레임워크 (1)

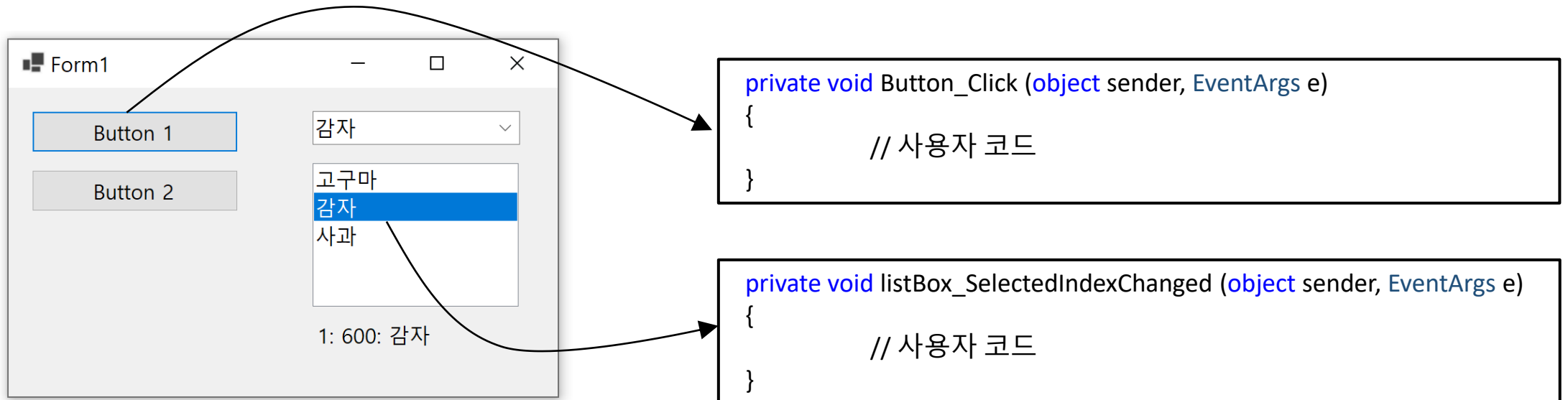
- 프레임워크 (Framework)
 - ✓ 제어 역전이 있는 대규모의 라이브러리
 - ✓ 응용 프로그램 개발에 바탕이 되는 템플릿과 같은 역할
 - ✓ 예: 윈도우 폼 (Windows Forms) 템플릿

```
1
2 namespace WinFormsApp1
3 {
4     참조 2개
5     partial class Form1
6     {
7         /// <summary>
8         /// Required designer variable.
9         /// </summary>
10        private System.ComponentModel.IContainer components = null;
11
12        /// <summary>
13        /// Clean up any resources being used.
14        /// </summary>
15        /// <param name="disposing">true if managed resources should be disposed; otherwise, false;
16        참조 1개
17        protected override void Dispose(bool disposing)
18        {
19            if (disposing && (components != null))
20            {
21                components.Dispose();
22            }
23            base.Dispose(disposing);
24        }
25    }
26}
27
28 #region Windows Form Designer generated code
29
```



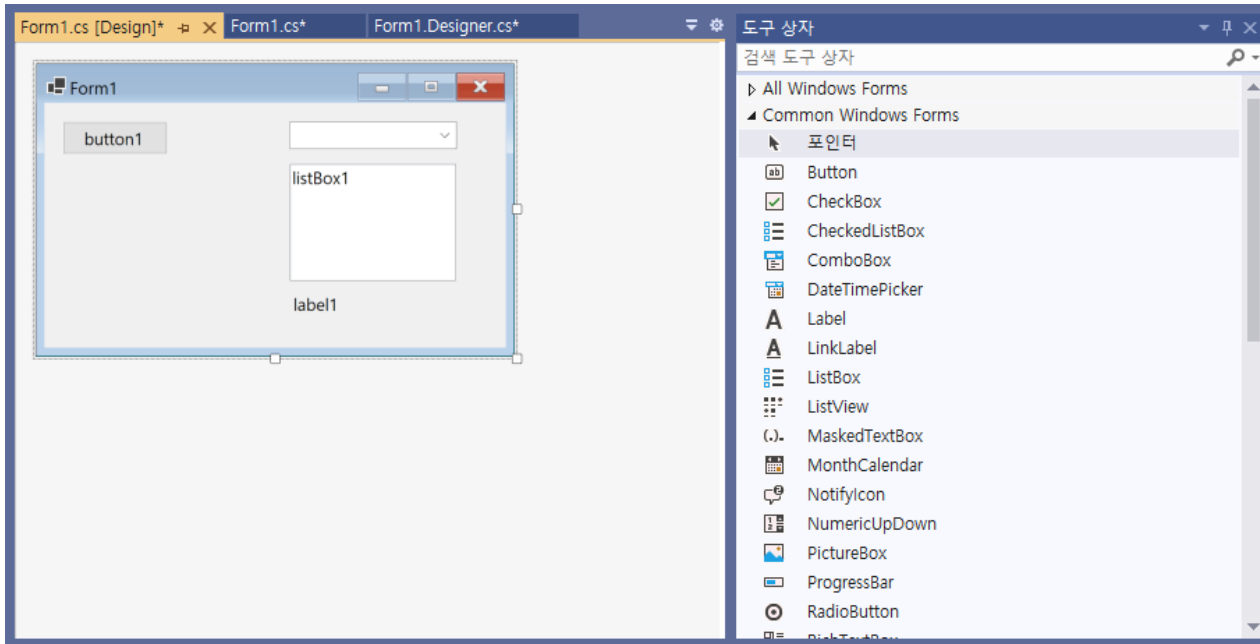
.NET Framework의 프레임워크 (2)

- 제어역전 (Inversion of Control: IoC)
 - ✓ 응용프로그램의 초기화부터 종료까지 흐름을 사용자 코드가 아닌 프레임워크에 의해 관리
 - ✓ 윈도 폼 예: 프레임워크는 필요 시에만 사용자 코드 호출 (e.g., 버튼 클릭 또는 리스트 박스 선택 요소 변경)



C# 프로그램 개발 – GUI (Graphical User Interface)

- 윈도우 폼 (Windows Forms)
 - ✓ 윈도우 UI (User Interface)를 가지는 프로그램
 - ✓ 기존 C++의 Win32 API 또는 MFC를 C#으로 옮겨 놓은 형태

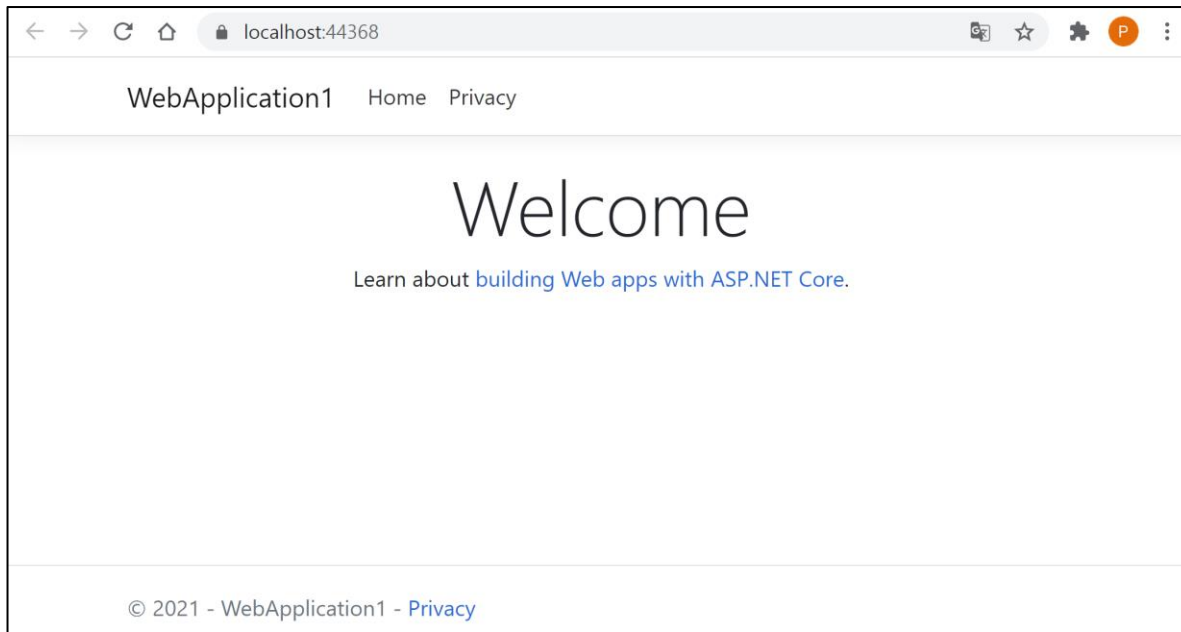


- ✓ 그림 그리듯 버튼 또는 콤보 박스와 같은 GUI 개발 가능
- WPF (Windows Presentation Foundation)
 - ✓ .NET 3.0부터 제공하는 GUI 프레임워크
 - ✓ 윈도우 폼보다 더 다양한 효과 제공



C# 프로그램 개발 – 웹 (Web)

- ASP.NET 웹 개발 프레임워크
 - ✓ ASP.NET 초기에는 주로 WebForms 사용
 - ✓ 현재는 테스트 및 유지 관리가 쉬운 ASP.NET MVC가 널리 사용

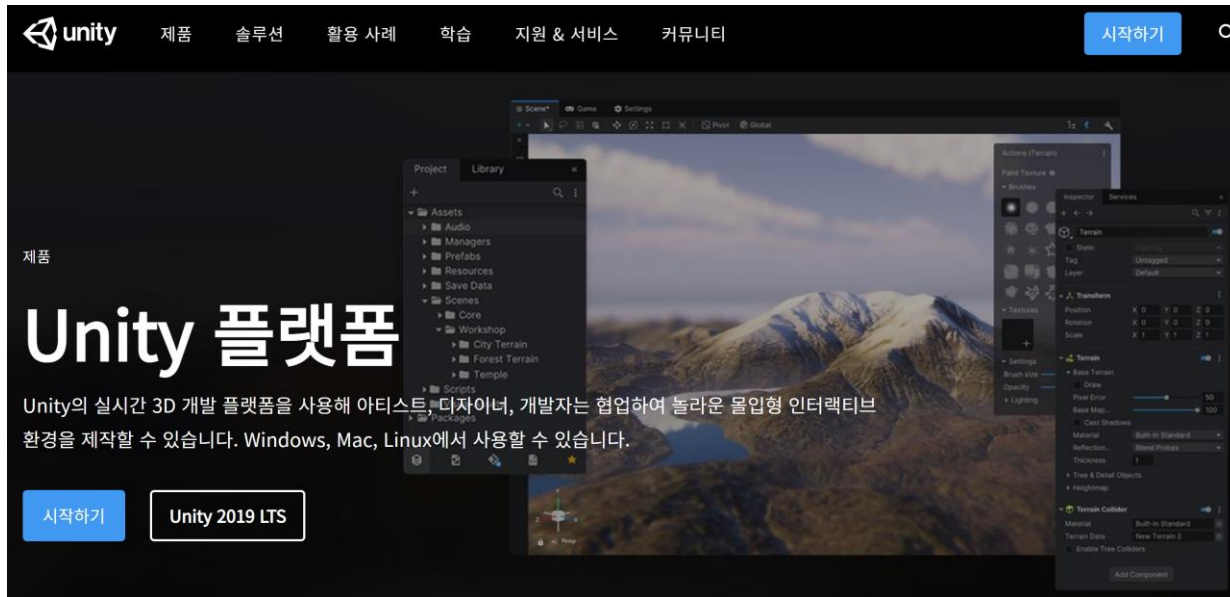


- ASP.NET MVC
 - ✓ **Model** : 데이터 및 로직 처리
 - ✓ **View** : 프로그램 외관, 결과 표현
 - ✓ **Controller** :
 - 사용자 입력/요청을 받음
 - Model를 통해 데이터 접근
 - View 생성하여 결과 리턴



C# 프로그램 개발 – 게임 (Game)

- 기존에는 데스크톱/콘솔 게임 개발을 위한 언어는 C++로 한정
- 최근 게임 서버/클라이언트 개발을 위해 C# 언어도 널리 사용

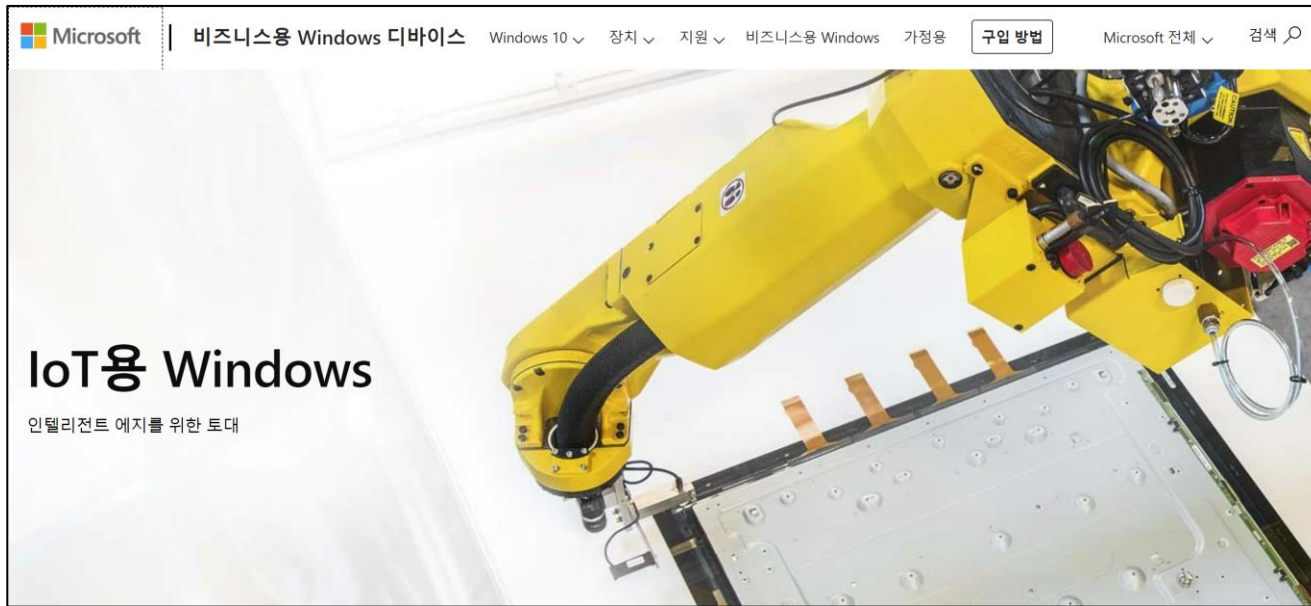


- 게임 클라이언트
 - ✓ 유니티 (Unity) 엔진 사용
 - ✓ 모바일/데스크톱/콘솔 개발
- 게임 서버
 - ✓ 많은 수의 사용자 지원
 - ✓ 다량의 데이터 실시간 전송 및 처리
 - ✓ 예 : MMORPG



C# 프로그램 개발 – IoT (Internet of Things)

- 사물 인터넷 (IoT)
 - ✓ 다양한 사물에 센서와 통신기능 내장, 인터넷 연결 기술
 - ✓ 은행 ATM 기계 또는 지하철 전광판 등에 활용



- Microsoft의 윈도우 10 IoT
 - ✓ 윈도우 임베디드 버전
 - ✓ 라즈베리파이 같은 싱글보드 컴퓨터에 활용 가능
 - ✓ C#을 이용해 IoT 앱 개발



C# 프로그래밍 시작하기

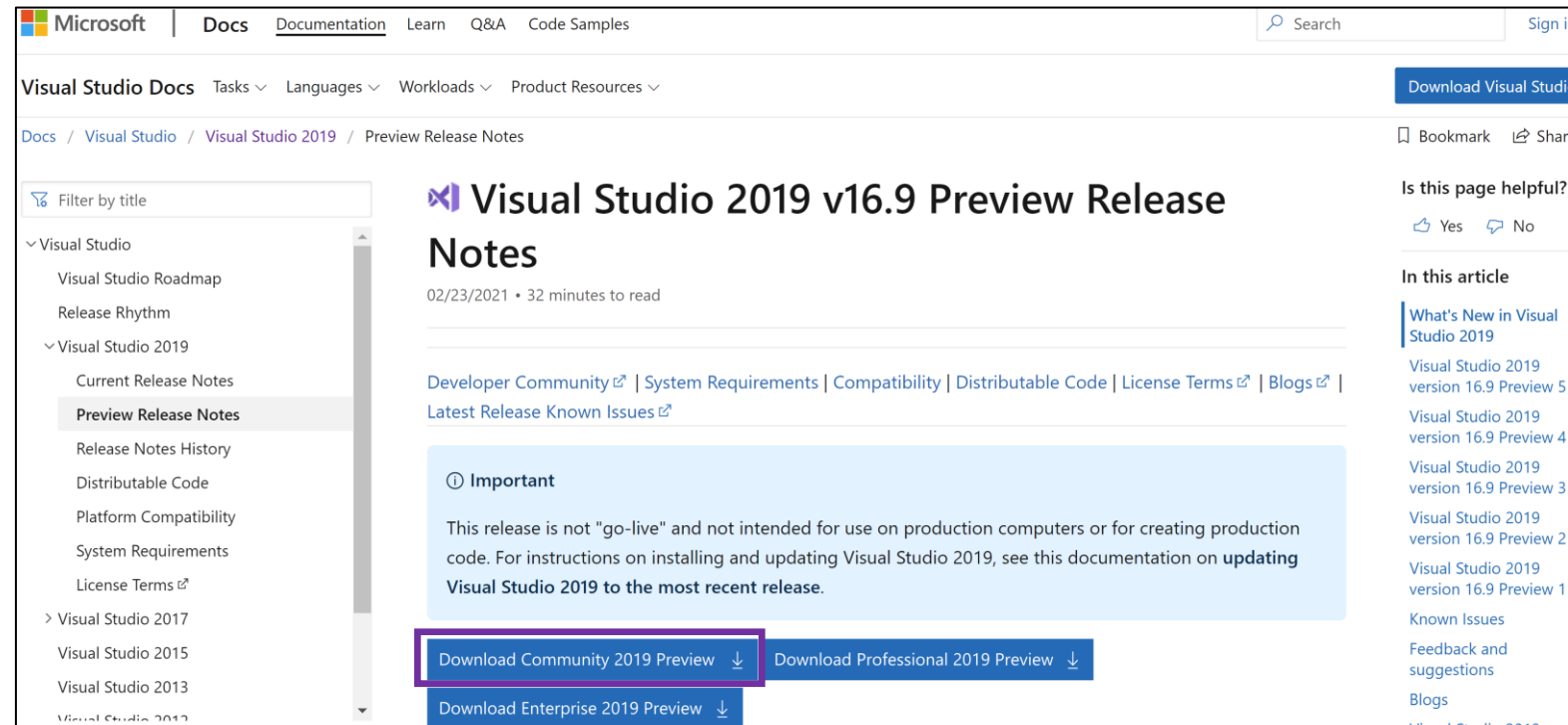
- C# 과 .NET 플랫폼

- 개발 도구 설치

- 예제 코드

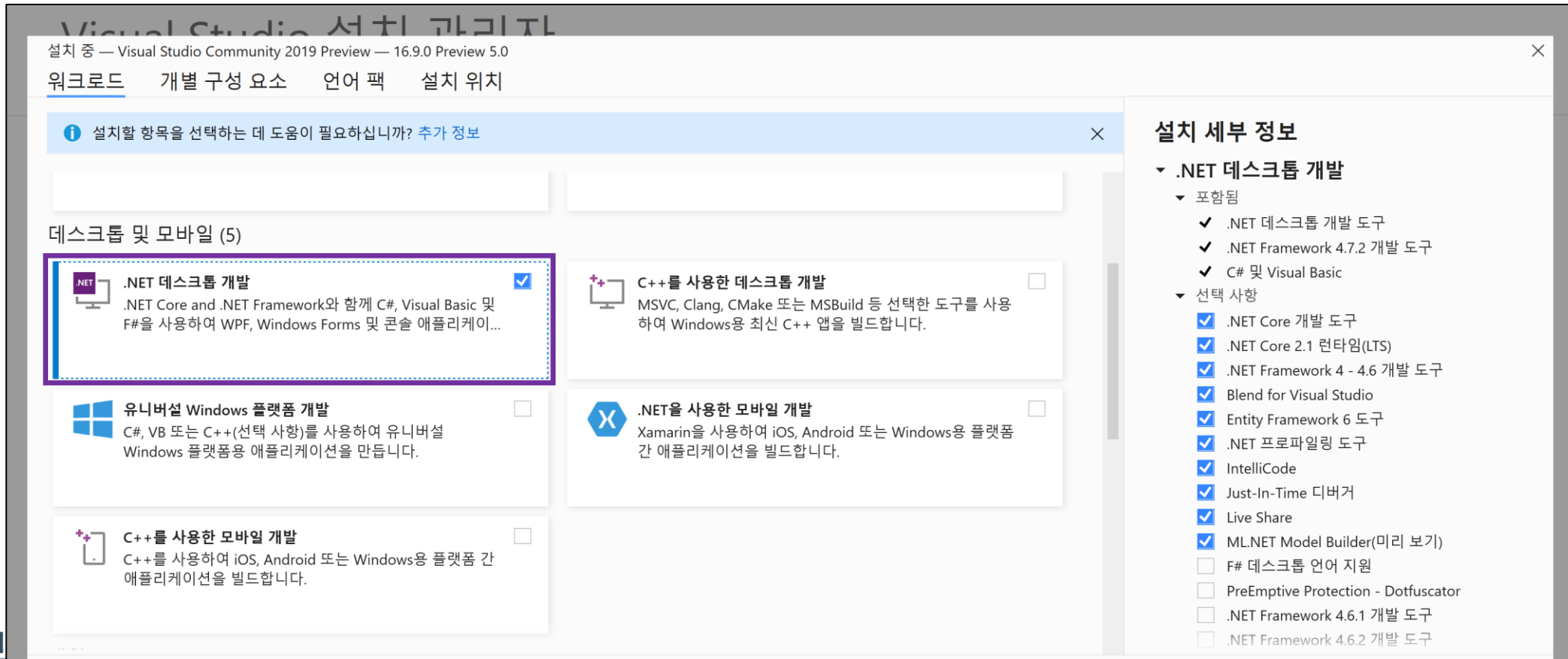
Visual Studio 2019 설치 (1)

- .NET 응용 프로그램 개발을 위한 가장 편리하고 빠른 환경 제공
- “Visual Studio 2019 v16.9 Preview 커뮤니티 버전” 다운로드
 - ✓ 링크 주소: <https://docs.microsoft.com/en-us/visualstudio/releases/2019/release-notes-preview#16.8.0.pre.5.0>



Visual Studio 2019 설치 (2)

- 다운받은 파일 더블 클릭하여 설치 프로그램 실행
 - ✓ 팩 설치 화면에서 "[데스크톱 및 모바일] - [.NET 데스크톱 개발]"에 체크



C# 프로그래밍 시작하기

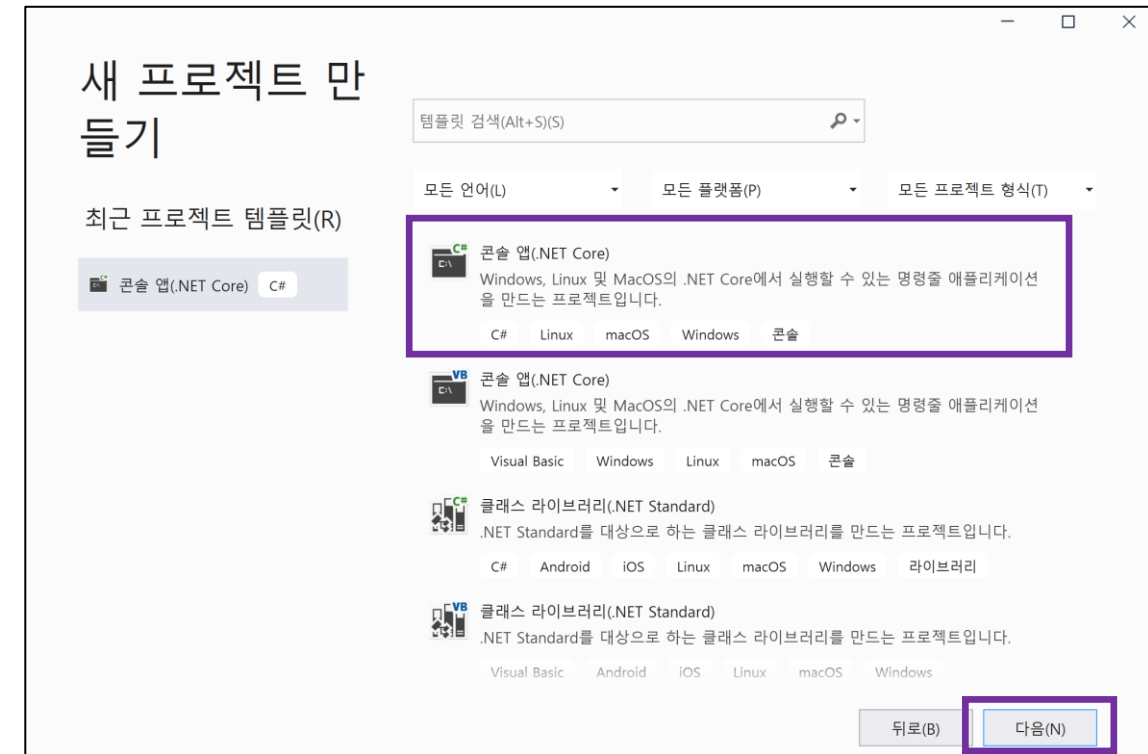
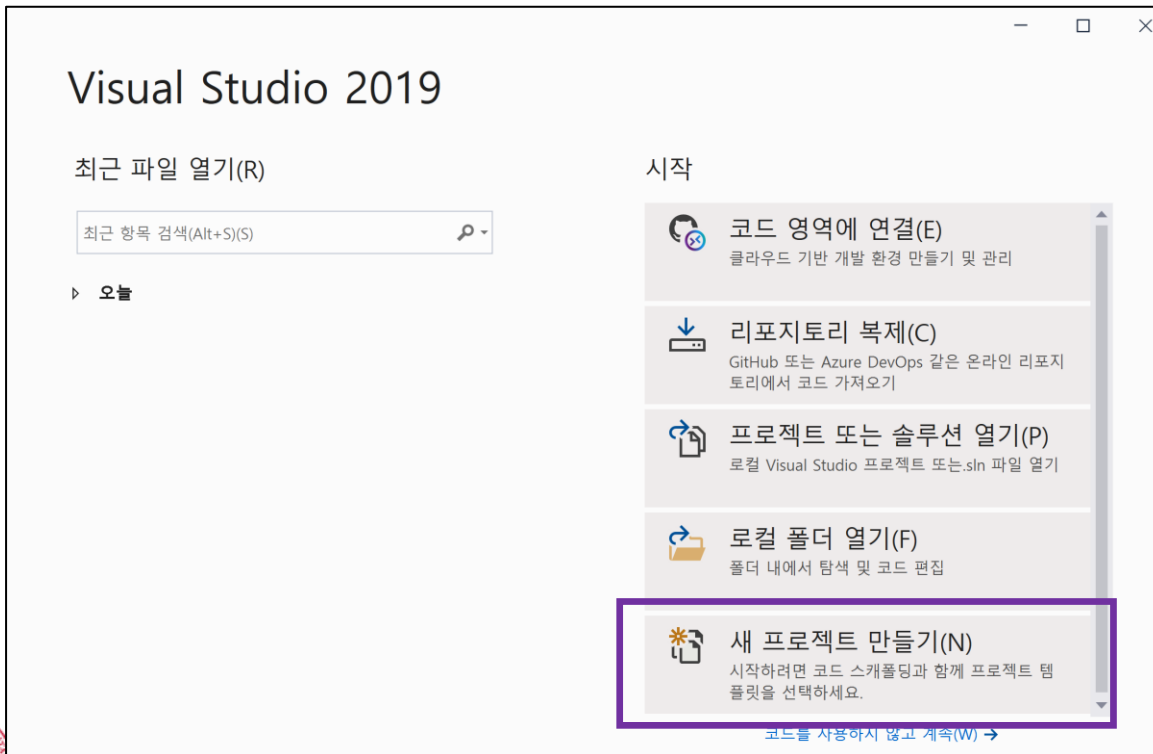
- C# 과 .NET 플랫폼

- 개발 도구 설치

- 예제 코드

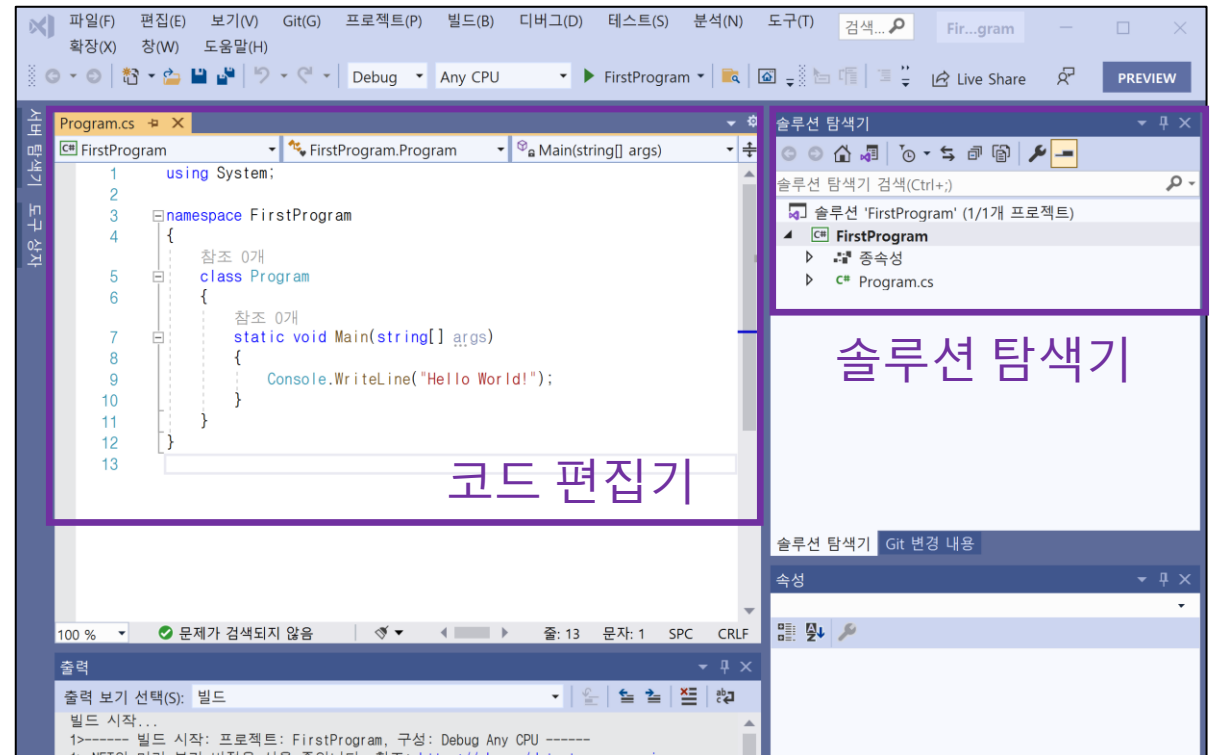
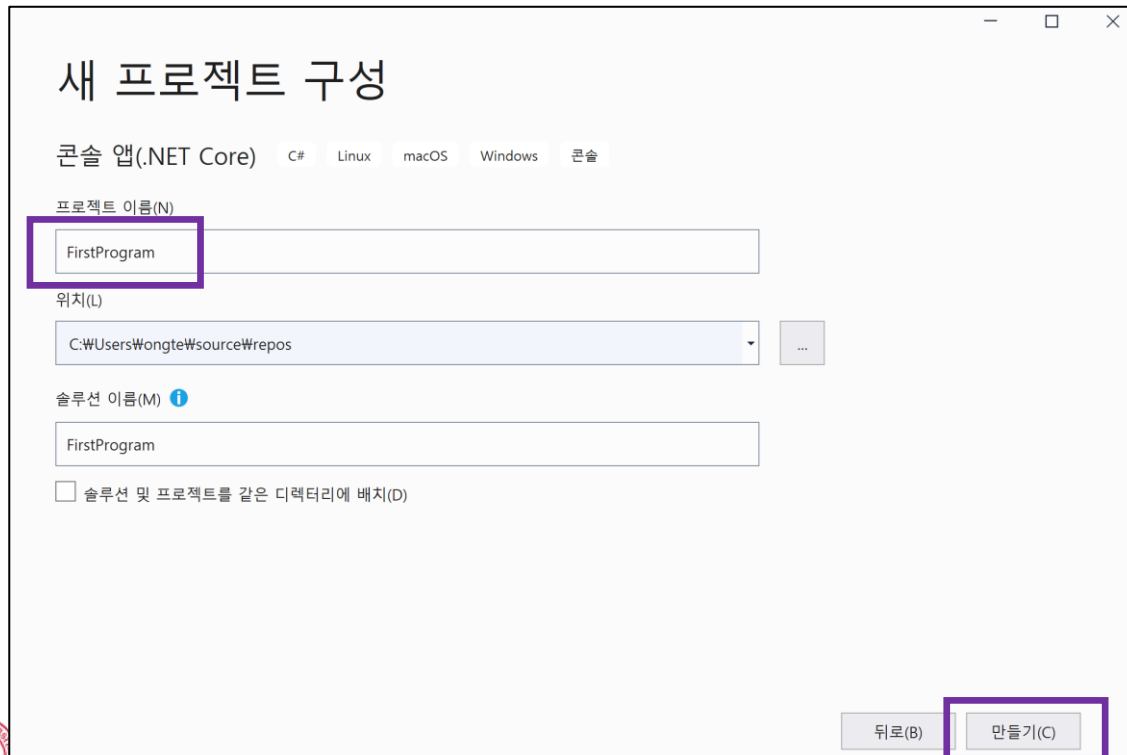
새 프로젝트 만들기 (1)

- Visual Studio 실행 후, <새 프로젝트 만들기> 버튼 클릭
- 템플릿 중에서 C#을 위한 [콘솔 앱(.NET Core)] 선택



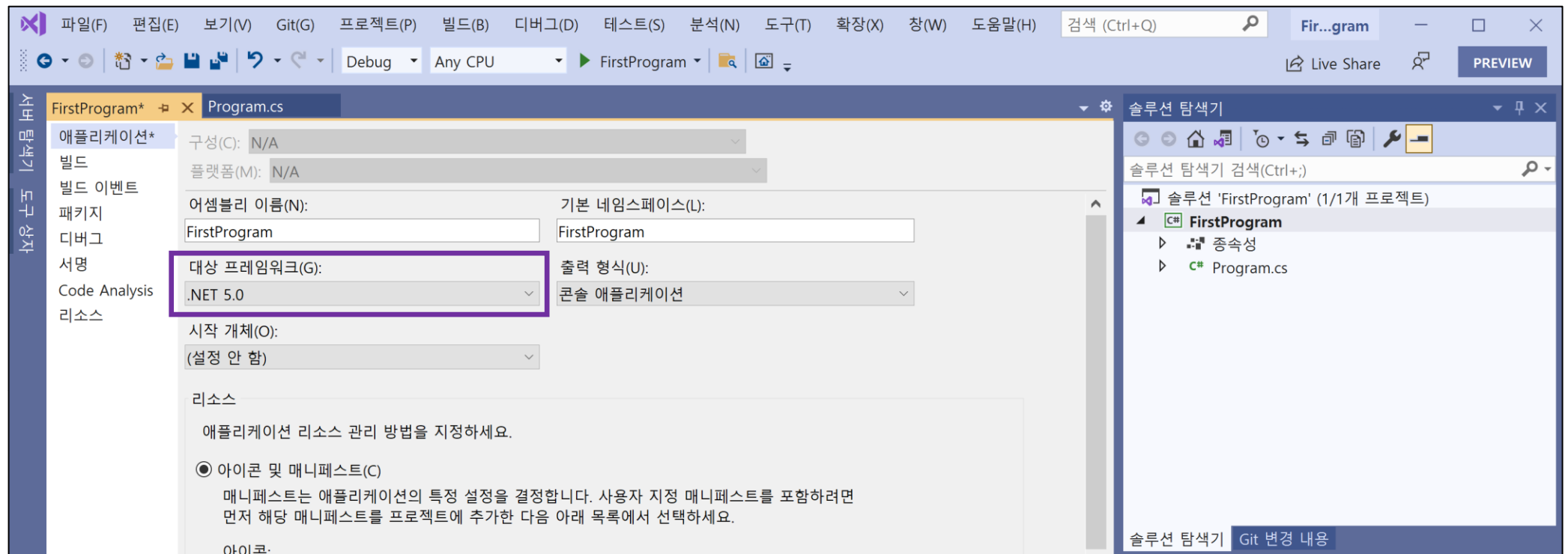
새 프로젝트 만들기 (2)

- “프로젝트 이름” 항목에 “FirstProgram” 입력 후 <만들기> 버튼 클릭
- 프로젝트 생성 후, 기본코드를 포함하는 코드 편집기와 솔루션 탐색기 생성
 - ✓ 솔루션 탐색기에서 확장자가 .cs 인 C# 소스파일 “Program.cs” 확인



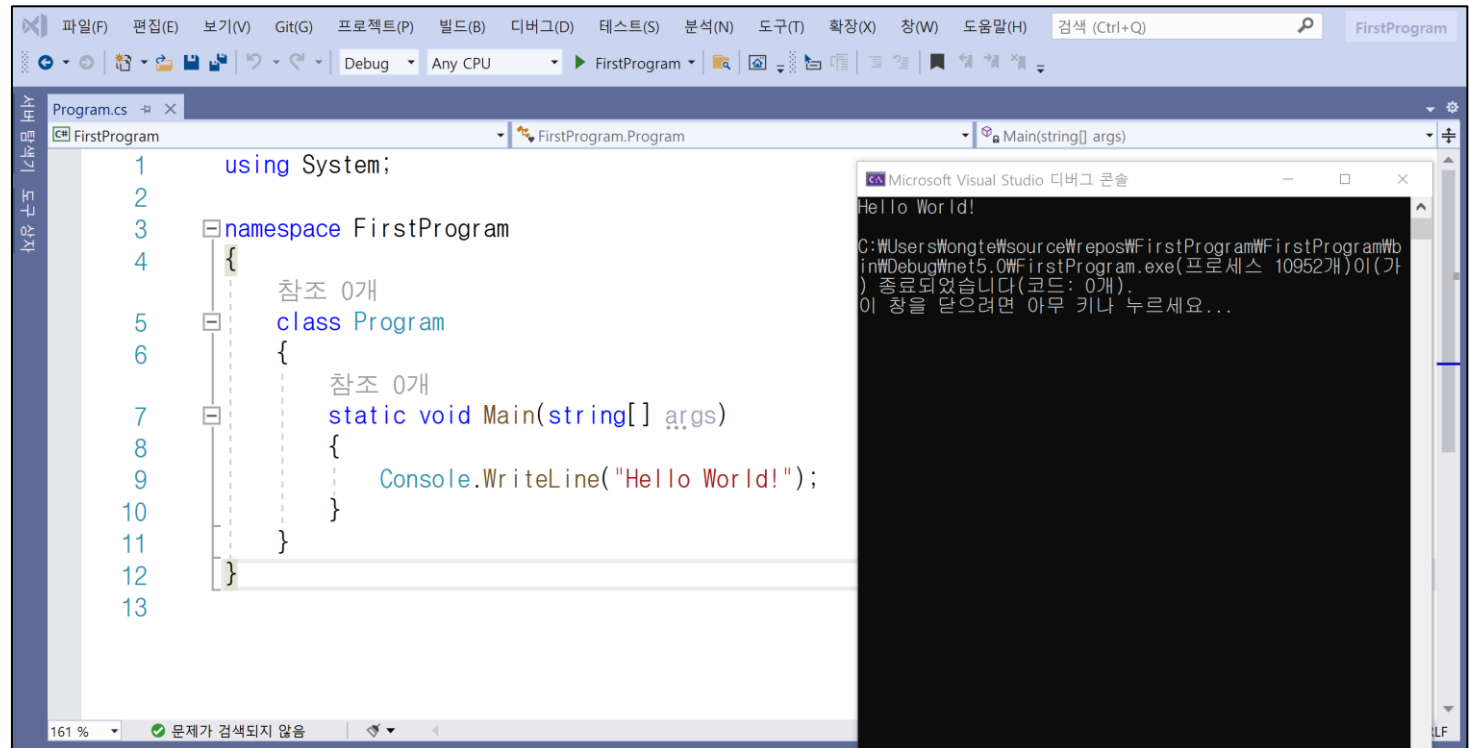
.NET 5.0 지정

- FirstProgram 프로젝트의 대상 프레임워크를 .NET 5.0 으로 설정
 - ✓ 메뉴의 [프로젝트] – [FirstProgram 속성]
 - ✓ [애플리케이션] 항목에서 [대상 프레임워크]를 “.NET 5.0”으로 지정



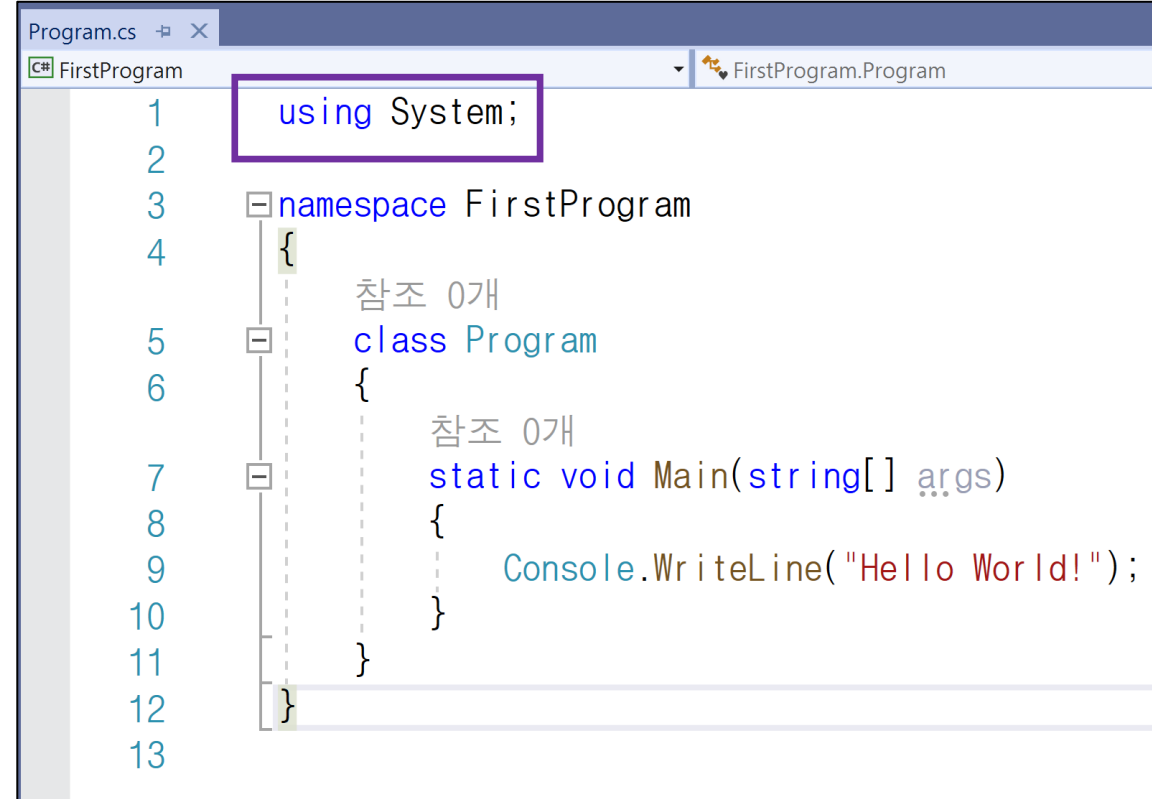
코드 실행

- 프로젝트 실행을 위해 키보드 "Ctrl + F5" 동시 입력
- 콘솔 화면에 "Hello World!" 실행 결과 출력



기본 소스코드 파악하기 (1)

- 기본 소스코드 해석을 통해 C# 프로그램의 골격 이해
- "using"
 - ✓ C# 언어 규격에 미리 정의 된 키워드
- "System"
 - ✓ C# 코드가 기본적으로 필요로 하는 클래스를 담고 있는 네임스페이스
 - ✓ "Console" 클래스 포함
- "using System;"
 - ✓ System 네임스페이스 내 클래스 사용을 컴파일러에 알리는 역할



```
Program.cs
C# FirstProgram
using System;

namespace FirstProgram
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Hello World!");
        }
    }
}
```

The screenshot shows a C# program in Visual Studio. The file is named 'Program.cs' and is part of a project named 'FirstProgram'. The code starts with the 'using System;' statement, which is highlighted with a purple box. This is followed by a namespace declaration 'namespace FirstProgram' and a class declaration 'class Program'. Inside the class, there is a static method 'Main' that takes an array of strings as an argument and calls 'Console.WriteLine' to print 'Hello World!'.



기본 소스코드 파악하기 (2)

- "namespace FirstProgram { }"

✓ 네임스페이스 생성

```
namespace 네임스페이스_이름
{
    // 클래스
    // 구조체
    // 인터페이스 등 ..
}
```

```
namespace FirstProgram
{
    class Program
    {
    }
}
```

- 다른 네임스페이스 내에서 Program 클래스 사용 방법

✓ "using FirstProgram;" 사용 또는

✓ "네임스페이스_이름.클래스_이름" 형태로 이용
(즉, FirstProgram.Program)



기본 소스코드 파악하기 (3)

- "class Program { }"
 - ✓ 클래스는 C# 프로그램을 구성하는 기본 단위
 - ✓ 클래스는 데이터와 데이터를 처리하는 기능(예: 메소드)로 구성
 - ✓ 기본코드에는 클래스 내 "Main()" 메소드 포함
- "static void Main(string[] args)"
 - ✓ 프로그램의 진입점 (Entry Point)
 - ✓ 프로그램 시작하면 실행, 이 메소드가 종료되면 프로그램도 종료
 - ✓ "static" 한정자 : static 키워드로 수식된 코드는 프로그램이 처음 구동될 때부터 메모리에 할당
 - ✓ "void" 메소드의 반환 형식
 - ✓ "string[] args" : 실행파일(FirstProgram.exe) 실행하는 명령어와 함께 입력되는 문자열 매개변수

```
class Program
{
    static void Main(string[] args)
    {
        Console.WriteLine("Hello World");
    }
}
```



The background of the slide is a dark blue gradient with a complex network of thin, light blue lines connecting small circular nodes. Some nodes are highlighted in green, and there are faint, larger geometric shapes in the background. A white rectangular border is centered on the slide, enclosing the text.

Thank you!